Int. J. Mach. Learn. & Cyber. (2017) 8:469–487
DOI 10.1007/s13042-015-0339-4

ORIGINAL ARTICLE

Online UAV path planning in uncertain and hostile environments

Naifeng Wen · Xiaohong Su · Peijun Ma · Lingling Zhao · Yanhang Zhang

Received: 17 June 2014 / Accepted: 6 February 2015 / Published online: 20 February 2015

Abstract Taking uncertainties of threats and vehicles' motions and observations into account, the challenge we have to face is how to plan a safe path online in uncertain and dynamic environments. We construct the static threat (ST) model based on an intuitionistic fuzzy set (A-IFS) to deal with the uncertainty of a environmental threat. The problem of avoiding a dynamic threat (DT) is formulated as a pursuit-evasion game. A reachability set (RS) estimator of an uncertain DT is constructed by combining the motion prediction with a RRT-based method. An online path planning framework is proposed by integrating a sub goal selector, a sub tasks allocator and a local path planner. The selector and allocator are presented to accelerate the path searching process. Dynamic domain rapidly-exploring random tree (DDRRT) is combined with the linear quadratic Gaussian motion planning (LQG-MP) method when searching local paths under threats and uncertainties. The path that has been searched is further improved by using a safety adjustment method and the RRT* method in the planning system. The results of Mont Carlo simulations indicate that the proposed algorithm behaves well in planning safe paths online in uncertain and hostile environments.

Keywords Online UAV path planning · Threat assessment · DDRRT · LQG-MP · A-IFS

N. Wen (✉) · X. Su · P. Ma · L. Zhao · Y. Zhang
School of Computer Science and Technology, Harbin Institute of Technology, Xidazhi Street, Harbin 150001, China
e-mail: wennaifeng@126.com

Abbreviations

A-IFS	An intuitionistic fuzzy set
ST	Static threat
DT	Dynamic threat
RS	Reachability set
DDRRT	Dynamic domain rapidly-exploring random tree
LQG-MP	Linear quadratic Gaussian motion planning
NFZ	No-fly zone
SH	Sensing horizon
PF	Particle filter
TS	Time stamp
IFWA	Intuitionistic fuzzy weighted averaging
TH	Time horizon
DD	Dynamic domain
TUDD	Threat and uncertainty based dynamic domain
CD	Collision detection

1 Introduction

UAVs often plan paths online in the low altitude to avoid threats by the masking of obstacles to threats in missions, e.g., anti-terrorist, motivating this study. A planner is also required to deal with uncertainties to further improve its abilities of avoiding threats and obstacles. In summary, online planning a path under uncertainties, complicated threats, dense obstacles and multi-constraints, becomes a practical challenge [1–8].

The computational time of the discretization methods grows exponentially as the problem dimensions increase [1–3, 9–11]. The randomized RRT plans possible paths to the goal by growing a path tree, under the guidance of random samples in the planning space [1–3]. It is not nearly encumbered by the dimensions and complexity of a

planning problem because its execution is independent of the environment [1–3, 12]. It checks constraints expediently [2]. It can be promoted by problem-specific heuristics [13, 14]. DDRRT is chosen to be the basis of our planning algorithm because it is efficient in obstacle-complex environments [15, 16]. The RRT-based estimators have advantages in both accuracy and efficiency when predicting the long-term possible DT paths under uncertain and limited information [17]. Thus, we propose a RRT-based RS estimation method with the consideration of the DT intention.

Uncertainties are inherent to the path planning problem. The methods, e.g., "relocalization", "landmark" and partially observable Markov decision processes, etc., useful for robot navigation under uncertainties, are difficult to be utilized online because of the heavy computational overheads [18]. Inspired by the simultaneous localization and mapping method [19], the works of path planning and uncertainties processing are executed simultaneously.

An environmental threat is constructed by a ST model and a motion model. Two traditional methods are often applied to construct the ST model. One regards a threat area to be impassable, simplifying the threat excessively. The other method utilizes a probabilistic model. Albeit it is practicable, it is coarse because the complicated model parameters cannot be considered thoroughly and accurately [5]. Besides, the probabilistic model has no special mechanism for expressing the uncertainty of a threat. Therefore, the A-IFS based ST model is constructed. Threats can be mainly classified into two groups, i.e., ST and DT. The difference between them is that DT has a motion model while ST does not have. Thus, ST is a subcase of the more general DT.

1.1 Dealing with uncertainties

Huynh et al. [20] proposed the "information constrained" linear quadratic Gaussian (icLQG) motion planning method to control a vehicle under imperfect information. An initial plan, including waypoints and the sensing utilities at the waypoints, is approximated first. Then feedback control inputs are iteratively computed to locally optimize the plan in the space of distributions over position, that is, in information space.

Van den Berg et al. [21] proposed the LQG-MP method to deal with the uncertainties of the motion and observation of a vehicle. They approximate the non-linear models of the motion and observation by a local linearization using the first-order Taylor expansion. Then the apriori probabilistic distributions of vehicle states are estimated by a linear quadratic regulator and the Kalman filter under Gaussian noises. During the estimation, the regulator provides online feedback control inputs for the Kalman filter which predicts the distributions of vehicle states. When generating paths, the linear quadratic Gaussian controller is applied as an optimizer. Finally, a set of candidate paths are created by RRT, and the best path is selected by the estimated distributions of vehicle states on waypoints. LQG-MP is a useful approach for planning a path under uncertainties. But the candidate paths are difficult to be calculated online because of the heavy computational overhead.

Based on the studies of [21–23], Jaillet et al. [24] presented the environment-guided RRT for dynamic and uncertain robot systems. The costs on waypoints are calculated by a LQG-MP based cost model.

Melchior et al. [25] raised the particle-RRT that extends a tree node multiple times according to the likely environmental conditions to decrease the environmental uncertainty. The newly generated nodes are regarded as particles. The likelihoods of particles are calculated by the distribution of the environmental conditions. Then particles are grouped to create waypoints and the highly likely paths are acquired. But the method is computational-complex.

Pepy et al. [26] regarded an uncertain configuration as a spheroid centering at the estimated configuration, shaped by the state distribution at the configuration. Burns et al. [27] applied the Bernoullian utility to calculate the path cost, to minimize the environmental uncertainty on a roadmap. Fraichard et al. [28] took both uncertainties and non-holonomic constraints into account when planning a path. Guibas et al. [29] evaluated the collision probability bounds on vertexes of a roadmap by the environmental uncertainty.

1.2 Threat assessment

Hanson et al. [30] created ST models by the fuzzy logic theory and assessed threats by the belief network. But the method is too complex to be utilized online. Kabamba et al. [31] constructed the probabilistic models of radar and missile.

Aoude and his colleague discussed the RS of DT is hard to be predicted by the traditional probabilistic methods when the DT motion is uncertain, flexible and intelligent without a long-term tendency [17]. Thus, they proposed the more suitable RRT-based estimators in [17, 32–36].

Reference [33] presented the RRT-Reach method to estimate the RS of an uncertain DT. RRT-Reach operates in two modes, i.e., exploration and pursuit. It operates as RRT in the exploration mode and as the greedy RRT in the pursuit mode. The Gaussian process is applied to predict the DT motion patterns to guide the sampling process in the pursuit mode. The DT avoidance problem is modeled as a pursuit-evasion game in the zero-sum differential game theory in [34]. The works' background is the intelligent transportation.

According to the traditional DT avoidance method based on the No-Fly Zone (NFZ) approach, if the occupied situation of a planning space by DT is identified, UAV will detour from the occupied spaces, as references [4, 37, 38] discuss. However, sometimes the threat areas is difficult to be avoided totally.

Karaman et al. [39, 40] proposed the RRT* method to find an asymptotic optimal path. The RRT* method firstly finds a path randomly. Then it replans and optimizes the path gradually by excluding non-optimal solutions. We regard the threats avoidance problem as an optimization problem. Thus, RRT* is directly reimplemented to plan a path under threats. However, the optimization process of RRT* is so slow that it may not be finished before the constraint time.

1.3 Assumptions and paper organization

We propose three background assumptions. a. UAV cannot avoid threats by simply increasing its speed or height. b. DT is a ground vehicle, which can adjust its path flexibly and intelligently with an intention of intercepting UAV. c. DT has the perfect knowledge of the UAV path planning.

The study is organized as follows. §2 describes the research problems mathematically. §3 proposes the threat assessment method. §4 offers an online path planning algorithm with the implementation and analysis. In §5, our simulation results are provided and discussed. In §6, conclusions are drawn.

2 Problem description

2.1 Online path planning under limited and uncertain information

We consider the following uncertainties. The threats on a possible UAV path are uncertain because they are also decided by the real-time UAV states. The threat information outside UAV Sensing Horizons (SHs) is unknown. The motions and observations of UAV and DT are uncertain. This is the first study which deals with the threats avoidance problem online under uncertainties.

The available information (I_t) includes: the distributions of UAV states calculated by the LQG-MP method; all observations (o) and states (s) at the time up to and including the present time (t); the past control inputs ($u \subseteq U$) before t where U is the set of allowable control inputs. The planning result is a feedback control sequence that can minimize the expected cost on a path from the start time 0 to an unspecific end time (T). The planning objective is similar to that in [20], expressed as: $\min_U \mathrm{C_u}(I_t) = \min_U \mathrm{E}\left[\int_0^T l(t, s_t, u(t, I_t)dt|I_t\right]$ where $l(t, s, u)$ denotes the instantaneous cost of the control input u_t at t.

The continuous planning can hardly be finished online. Hence, it is approximated by a discrete-time model: $\min_U \mathrm{C_u}(I_k) = \min_U \mathrm{E}\left[\sum_{k=0}^{N} l_k(s_k, u_k)|I_k\right]$. N is the number of time steps, $N = T/\tau$. τ is the duration of one control time step.

2.2 UAV motion model and dynamic constraints

The equations of motion and observation of UAV are defined as follows:

$$\mathrm{f}(s,u,m) = \begin{bmatrix} x + \tau v \cos\varphi \cos\theta \\ y + \tau v \cos\varphi \sin\theta \\ z + \tau v \sin\varphi \\ \theta + \tau(\omega_\theta + \tilde{\omega}_\theta) \\ \varphi + \tau(\omega_\varphi + \tilde{\omega}_\varphi) \\ v + \tau(a + \tilde{a}) \end{bmatrix},$$

$$\mathrm{h}(s,n) = \begin{bmatrix} x_k + \tilde{x} \\ y_k + \tilde{y} \\ z_k + \tilde{z} \\ \arctan(\bar{y}/\bar{x}) + \tilde{\theta} \\ \arctan(\bar{z}/\sqrt{\bar{x}^2 + \bar{y}^2}) + \tilde{\varphi} \end{bmatrix}$$

The UAV state vector s is defined as $[x, y, z, \theta, \varphi, v]^{\mathrm{T}}$ where $[x, y, z]$ denotes a location of UAV, θ and φ denote the yaw angle and pitch angle respectively, $v \subseteq [\mathrm{v_{min}}, \mathrm{v_{max}}]$ denotes a velocity. The control input $u = (a_v, \omega_\theta, \omega_\varphi)$ incorporates an acceleration (a_v), a yaw steering (ω_θ) and a pitching motion (ω_φ). u is corrupted by the process noise $m = (\tilde{a}_v, \tilde{\omega}_\theta, \tilde{\omega}_\varphi) \sim \mathrm{N}\left(0, \begin{bmatrix} \sigma_{a_v}^2 & 0 & 0 \\ 0 & \sigma_{\omega_\theta}^2 & 0 \\ 0 & 0 & \sigma_{\omega_\varphi}^2 \end{bmatrix}\right)$. The turning radius ($r$) is bounded by the minimum turning radius, $r \geq \mathrm{r_{min}}$. The straight path length (sl) before turning is bounded by $\mathrm{sl_{min}}$, $sl \geq \mathrm{sl_{min}}$.

$\mathrm{h}(s,n)$ is the observation equation. $\bar{x} = x_k - x_{k-1}$, $\bar{y} = y_k - y_{k-1}$, $\bar{z} = z_k - z_{k-1}$. The observation noise $n = (\tilde{x}, \tilde{y}, \tilde{z}, \tilde{\theta}, \tilde{\varphi}) \sim \mathrm{N}\left(0, \begin{bmatrix} \sigma_x^2 & 0 & 0 & 0 & 0 \\ 0 & \sigma_y^2 & 0 & 0 & 0 \\ 0 & 0 & \sigma_z^2 & 0 & 0 \\ 0 & 0 & 0 & \sigma_\theta^2 & 0 \\ 0 & 0 & 0 & 0 & \sigma_\varphi^2 \end{bmatrix}\right)$.

Dynamic constraints often arise from the set of control inputs [1–3]. They restrict the allowable velocities at each waypoint on a path. Since the constraints are complicated, the state transitions via control inputs are often difficult to

be parameterized. However, the RRT-based methods can directly handle the constraints by the piecewise constraint-checking approach. A path tree extension can be executed only if it satisfies all constraints. Thus, all the sub paths or branches on the path tree are surely feasible.

2.3 Dynamic threat avoidance

The DT avoidance problem is formulated as a zero-sum differential game with multi-constraints, free final time and perfect UAV path planning information. The formulation derives from [41, 42]. Let t_f be the estimated earliest time DT threatens UAV. $t_f = +\infty$ indicates that UAV can reach goal without DT threat. $t_f = 0$ means DT has already threatened UAV. UAV aims to maximize t_f while DT aims to minimize t_f. We assume that only one DT and one UAV participate in the game (G) to clearly describe the DT avoidance problem. G terminates when UAV reaches goal with a practical safe and feasible path.

Let the subscript 1 and 2 denote DT and UAV respectively. The payoff function of G is expressed as $\mathrm{J}(s_t, u_t, t) = t_f$ where $s_t \subseteq S$ denotes the state of DT or UAV at the time t, $u_t \subseteq U$ is the control input of DT or UAV with constraints. We suppose that G admits a saddle point $(\gamma_1^*, \gamma_2^*) \subseteq U_1 \times U_2$ in the set of feedback control inputs. If both players execute that control strategy, the optimal payoff function value is $\mathrm{J}(s_t, t)^* = \max_{r_2^* \subseteq U_2} \min_{r_1^* \subseteq U_1} \{\mathrm{J}(s_t, u_t, t)\}$.

We cannot consider the full set of control inputs of DT and UAV online due to the high computational overhead. Thus, the feasible sets of control inputs $\tilde{U}_1 \subseteq U_1$ and $\tilde{U}_2 \subseteq U_2$ are approximated. The objective payoff function value is approximate by a practical one: $\tilde{\mathrm{J}}(s_t, t)^* = \max_{r_2^* \subseteq \tilde{U}_2} \min_{r_1^* \subseteq \tilde{U}_1} \{\mathrm{J}(s_t, u_t, t)\}$.

3 Threat assessment

3.1 Static threat model construction

We construct the ST model based on A-IFS to deal with the uncertainty of a threat. A-IFS is characterized by a membership function, a non-membership function and a hesitancy function [43]. It deals with the vagueness flexibly according to the human empirical knowledge. It also helps to extract more information from data. It provides plenty of rules and aggregation operators to quantize and aggregate information without any loss [43].

Let X be a fixed set. The A-IFS A in X is defined as: A is a nonempty set, $A = \{\langle a, \mu(a), v(a)\rangle | a \subseteq X\}$, $\mu(a)$ and $v(a)$ are the functions of membership and non-membership of A, $\mu(a) : X \to [0,1], a \subseteq X \to \mu(a) \subseteq [0,1], v(a) : X \to [0,1]$, $a \subseteq X \to v(a) \subseteq [0,1]$, $\mu(a) + v_A(a) \leq 1$. $\pi(a) = 1 - \mu(a) - v(a)$ is the hesitancy degree of a to A or the intuitionistic index of a in A.

The membership function is defined as the active threat function of a threat. The nonmembership function is defined as the threat-free function of UAV according to the real-time states of UAV. Moreover, a certainty function is proposed to deal with the uncertainty of a threat.

We take radar for an example to describe the processes of the ST model construction and threats avoidance. The height of the radar detection boundary $h_B = \mathrm{K_R} \cdot L^2$. $\mathrm{K_R}$ is the radar characteristic coefficient. L is the horizontal distance from the radar center. The area below h_B is termed as the radar blind spot [44] which can be utilized by UAV. The static model of DT is the same as the radar model except its h_B is set to be zero. The radar membership function is defined according to a basic damage probabilistic model in [44], as:

$$\mu_{\mathrm{D}} = \begin{cases} 0 & h < h_B \\ e^{-\frac{R^4}{\mathrm{R}_{\max}^4}} & h \geq h_B \end{cases} \tag{1}$$

where R denotes the radial distance from the radar center. $\mathrm{R_{max}}$ denotes the maximum threat radius. We only consider the threat within the $\mathrm{R_{max}}$.

UAV can reduce risks by applying strategies, i.e., turning to fly away from threats and increasing its speed. Thus, two parameters are considered in the non-membership function: v_T denotes the speed of UAV, $\alpha_T (0 \leq \alpha_T \leq \pi)$ denotes the absolute value of the angle between the direction of the velocity and the line from UAV to the center of a threat. Accordingly, the non-membership function is defined as:

$$\mathrm{V_D} = \begin{cases} 0 & v_{min} \leq v_T < v_p, \quad 0 \leq \alpha_T \leq \pi \\ \sin\left(\dfrac{\pi(v_T - \mathrm{v_{min}})}{2(\mathrm{v_{max}} - \mathrm{v_{min}})}\right) \sin^2\left(\dfrac{\alpha_T}{2}\right) & v_p \leq v_T < \mathrm{v_{max}}, \quad 0 \leq \alpha_T \leq \pi \end{cases}$$

where v_p is the threshold of the UAV penetration speed.

The certainty function of the radar model is defined as:

$$\mathrm{CF} = 1 - \left(\frac{1}{\omega_N N_e + \omega_\delta \delta_q}\right)^{\omega_C} \tag{2}$$

where N_e denotes the number of threats in which UAV exposes. δ_q denotes the standard deviation of the UAV state distribution at a candidate waypoint q estimated by LQG-MP. The weights $\omega_N + \omega_\delta = 1$. ω_C aims to bound the influence of CF and make $\mathrm{CF} \leq 1$. CF is defined to reduce N_e and δ_q to decrease the threat and increase the control certainty on a path.

We weight $[\mu_{\mathrm{D}}(a), \mathrm{V_D}(a)]$as: $[\mu_{\mathrm{D}}{}'(a), \mathrm{V_D}{}'(a)]^{\mathrm{T}} = \mathrm{CF}(a) \cdot [\mu_{\mathrm{D}}(a), \mathrm{V_D}(a)]^{\mathrm{T}}$. We make conservative assessment of a threat to promote the safety of a path, i.e., $\mu_{\mathrm{D}}'(a)$

has a priority over $V_D^{'}(a)$. If $\mu_D^{'}(a) = 1\ or\ 0$, set $V_D^{'}(a) = 0$. If $\mu_D^{'}(a) + V_D^{'}(a) > 1$, a normalization is executed to make $\mu_D^{'}(a) + V_D^{'}(a) = 1$. The score function of A-IFS is employed to calculate the threat level of a, i.e., $Tl(a) = \mu_D(a) - V_D(a)$, if $V_D(a) > \mu_D(a)$, set $Tl(a) = 0$. If the threat levels of two threats are equal, the threat with the bigger CF value is more dangerous.

3.2 Reachability set estimation

The DT assessment is the same as the ST assessment given the RS. The RS estimation incorporates the processes of motion prediction and RS searching. The DT motion predictions are applied to guide the path searching process.

3.2.1 Dynamic threat motion prediction

The particle filter (PF) [45] is utilized for the DT motion prediction. The equations of the states transition and the observation of DT are defined as:

$$s_k = \begin{bmatrix} 1 & \tau & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & \tau \\ 0 & 0 & 0 & 1 \end{bmatrix} s_{k-1} + \begin{bmatrix} \tau^2/2 & 0 \\ \tau & 0 \\ 0 & \tau^2/2 \\ 0 & \tau \end{bmatrix} \omega_k,$$

$$o_k = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} s_k + \begin{bmatrix} \sigma_x^2 & 0 \\ 0 & \sigma_y^2 \end{bmatrix} \upsilon_k$$

A DT state is denoted by $s = [x, \dot{x}, y, \dot{y}]^T$ where $[x, y]$ denotes a position, $\dot{x}$ and $\dot{y}$ denote the velocities in the x and y directions, respectively. ω_k and υ_k are white Gaussian noises, $\omega_k = \begin{bmatrix} \omega_{xk} \\ \omega_{yk} \end{bmatrix} \sim N\left(0, \begin{bmatrix} \sigma_{\omega_x}^2 & 0 \\ 0 & \sigma_{\omega_y}^2 \end{bmatrix}\right)$, $\upsilon_k \sim N(0, 1)$.

PF estimates the distributions of the DT states by approximating the discrete positions and weights of particles. The PF iteration is as follows.

1. A set of initial particles $\{p_0^i\}_{i=1}^{N_s}$ are randomly and uniformly sampled around the initial DT position. The weights of particles are set to be $1/N_s$.
2. The particles $\{p_k^i\}_{i=1}^{N_s}$ in the kth time step are predicted from $\{p_{k-1}^i\}_{i=1}^{N_s}$ by the states transition equation. Their weights $\{\omega_k^i\}_{i=1}^{N_s}$ are calculated subject to $\{\omega_k^i\}_{i=1}^{N_s} = \{\omega_{k-1}^i\}_{i=1}^{N_s} L(o_k'|s_k)$. $L(o_k'|s_k)$ is the likelihood function reflecting the similarity between s_k and the actual observation o_k'. We express the observation function by $o_k = Hs_k + Qv_k$. Then we calculate the weights of the particles in the kth iteration by $p(w_k^i) = \frac{1}{2\pi\sqrt{\det(Q)}} e^{-(o_k' - Hs_k)^T Q^{-1} (o_k' - Hs_k)}$ where $\det(Q)$ is the determinant of Q.
3. The weights are normalized. New particles are generated by "resampling" from the obtained particles to avoid the particle degeneration. The DT state is predicted by the mean of the resampled particles. Go back to step 2.

In the resampling process, the particles with higher weights or higher probabilities to be corrected have higher probabilities to be kept. The iteration is as follows. Firstly, a uniformly and randomly distributing number $\xi_i \in [0, 1]$ is created. Then the first particle with the cumulative weights $a_{\omega_j} \geq \xi_i$, $a_{\omega_j} = \omega_1 + \omega_2 + \ldots + \omega_j$ is returned. The loop is executed N_s times to make N_s new particles.

3.2.2 Reachability set searching

The RS of an uncertain DT is searched by the RRT-based method under imperfect motion data. A time stamp (TS) is explicitly added on each node of the DT or UAV path trees as a kind of time heuristics. A TS is the predicted earliest time of a vehicle arriving at a tree node based on the Dubins distance. TS helps our method to identify low time paths and approximate a more realistic RS as well as combining a path with time.

Figure 1 shows the DT path tree (*TTree*) updating by the guidance of a new observation. The crosses denote the observed positions of DT. The blue points denote the particles in PF. The red dotted line denotes a predict DT path. The root of *TTree* is relocated to the new observation. An extension is made from the new *TTree* root along the predicted path. A new *TTree* node $TNode_{new}$ is created as the unique child of the root. The outdated nodes of which the TSs are less than or equal to the TS of $TNode_{new}$ are deleted, so do the relevant edges, as the black nodes and dash-dot lines show. Then we reconnect the remaining parts of *TTree* through $TNode_{new}$ and update the TSs of all the *TTree* nodes.

A reachable node is defined as a UAV path tree (*PathTree*) node that DT can reach by a possible path. Fig. 2 shows the interpretation of DT reached q_i. The red-node and blue-node trees denote *TTree* and *PathTree*, respectively. We check whether DT is reachable to q_i through $TNode_j$ as follows.

1. The horizontal projection point ($HP(q_i)$) of q_i is computed. The horizontal boundary ($TB(q_i)$) of the DT threat around $HP(q_i)$ is calculated depending on

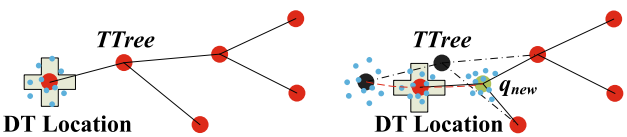


Fig. 1 DT path tree updating by the guidance of a new observation

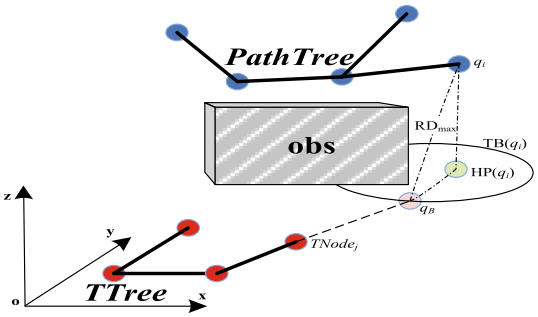


Fig. 2 Interpretation of DT being reachable to q_i

the maximum threat radius (RD_{max}) of the static DT model.

2. We search for a point q_B on $TB(q_i)$, satisfying: q_B is a point on $TB(q_i)$; the line between q_B and q_i is obstacle-free; the line between $TNode_j$ and q_B is obstacle-free; the estimated time DT reached q_B is less than the TS of q_i.

DT is reachable to q_i through $TNode_j$ if such q_B is found. If DT is reachable to q_i, it may also threaten the neighbors of q_i. Thus, the estimator continues searching for reachable nodes among the neighbors on the UAV path tree. We call q_i as the directly DT reachable node, the reachable neighbors as the indirectly reachable nodes. The estimated shortest distances between DT and its reachable nodes are computed to assess the DT threats on the nodes.

We extend four RRT-based RS searching operators to rapidly approximate the RS. Each possible DT path on its path tree is regarded as a possible DT motion pattern or strategy. Fig. 3 shows the four target-bias operators. "goal-bias-connect" is the sub operator of the other three operators. "multi-reach" is the sub operator of "multi-connect". The dotted lines denote the pursuit directions of *TTree* nodes to *PathTree* nodes.

"Connect" and "Greedy-Connect" are two basic RRT extension operators. $\text{Connect}(Tree, q_n, q_s)$ extends q_n one step further to q_s. The extension function is $(u, q_{new}, e_{new}) = \text{Extend}(q_n, le, q_s, U)$ where q_{new} and e_{new} are the newly created tree node and branch, respectively. $u \in U$ is the control input on e_{new}, le is the length of the extension. $\text{Greedy} - \text{Connect}(Tree, q_n, q_s)$ extends q_n steps towards q_s by "Connect" greedily, until the extension reaches q_s or the extension is blocked by obstacles.

1. $\text{goal} - \text{bias} - \text{connect}(q_i, TNode_j, TTree)$ is executed as follows. If DT is reachable to a *PathTree* node (q_i) through q_i's nearest *TTree* node ($TNode_j$), then $\text{Greedy} - \text{Connect}(TTree, TNode_j, q_B)$ is executed. q_B is the same as that in Fig. 2. The indirectly reachable nodes is searched among the neighbors of q_i on

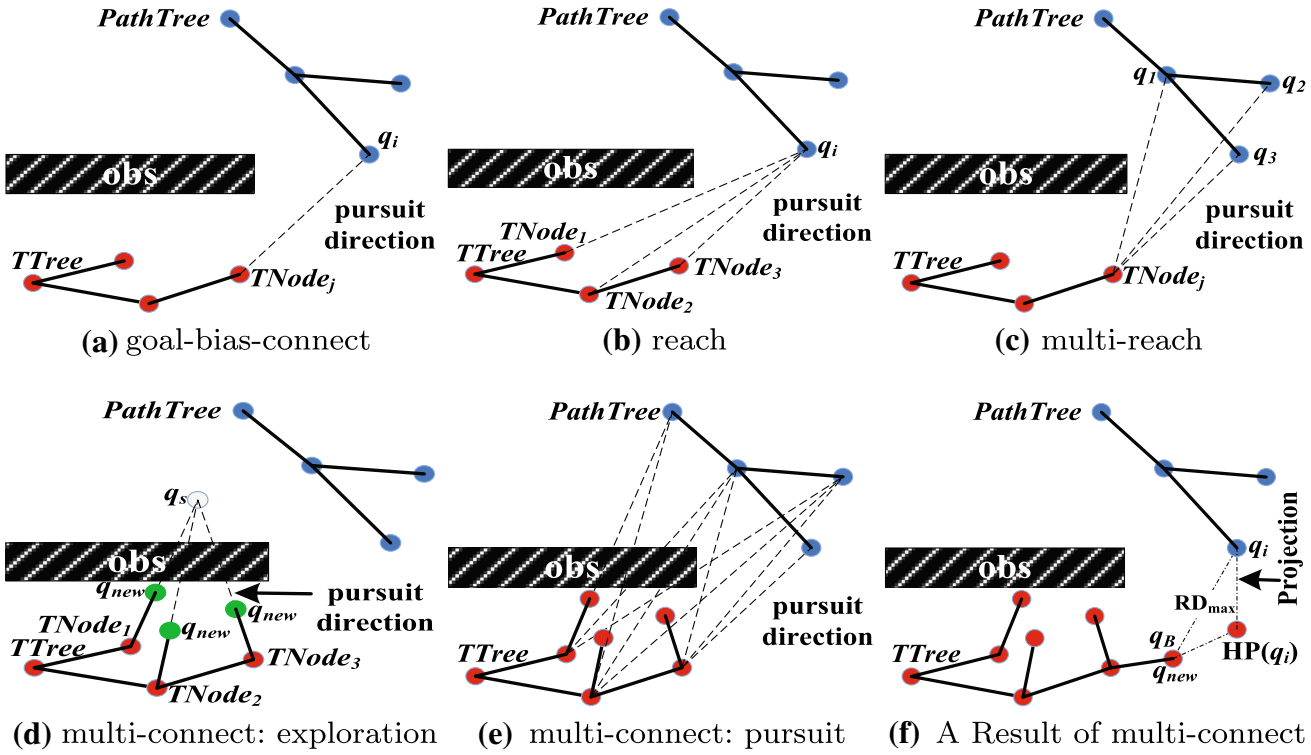


Fig. 3 RS searching operators with target-bias based on RRT

PathTree. Otherwise, Connect$(TTree, TNode_j, q_B)$ is used to explore the planning space.

2. In reach$(q_i, TTree)$, the k-near *TTree* nodes $(TNode_j, 1 \le j \le k)$ of q_i are found and sorted by their TSs. goal − bias − connect$(q_i, TNode_j, TTree)$ are executed in turn until *TTree* reaches q_i by an extension or all $TNode_j$ have been tried. Fig. 3 is an illustration when $k = 3$.
3. In multi − reach$(TNode_j, PathTree, TTree)$, the k-near *PathTree* nodes $(q_i, 1 \le i \le k)$ of a *TTree* node $(TNode_j)$ are found and sorted by their TSs. Then, goal − bias − connect$(q_i, TNode_j, TTree)$ are executed in turn until a q_i is reached by DT or no q_i is reachable. The operator aims to increase the probability of capturing reachable nodes through a *TTree* node.
4. multi − connect$(q_s, TTree, PathTree)$ is designed to enlarge RS and keep the diversity of RS by the guidance of random samples. It executes in two steps, i.e., exploration and pursuit. In the exploration step, a random sample q_s is generated and the k-near *TTree* nodes $(TNode_j\ 1 \le j \le k)$ of q_s are found and sorted by the TSs, then Connect$(TTree, TNode_j, q_s)$ are executed, as Fig. 3d shows. In the pursuit step, multi − reach$(TNode_j, PathTree, TTree)$ are executed in turn, until DT is reachable to a q_i or all $TNode_j$ have been tried, as Fig. 3e shows. Fig. 3f shows a possible result of "multi-connect" and q_{new} is identical with q_B. RD_{max} and $HP(q_i)$ are the same as those in Fig. 2.

"multi-connect" utilizes "Connect" in the exploration step. "Connect" has the guarantee of probabilistic completeness [1, 2]. Thus, "multi-connect" operator is also probabilistic complete. However, computing the full RS online is computationally complex. Thus, "multi-connect" and "reach" are called randomly subject to a certain probability threshold. "goal-bias-connect" and "multi-reach" do not run independently but being executed as sub operators.

3.3 Threats aggregation

The DT assessment is similar to the ST assessment, given the RS. The threats aggregation is required to both calculate an average threat level and emphasize the roles of high risk threats [43]. Thus, the Intuitionistic Fuzzy Weighted Averaging (IFWA) operator [43] is utilized due to its satisfaction of the requirement. $\mathrm{IFWA}_\omega(a_1, a_2, \ldots, a_n) = \omega_1 a_1 \oplus \omega_2 a_2 \oplus \ldots \oplus \omega_n a_n = \left(1 - \prod_{i=1}^{n}\left(1 - \mu_{a_i}\right)^{\omega_i}, \prod_{i=1}^{n} v_{a_i}^{\omega_i}\right)$ where $\oplus$ is an A-IFS operator, ω_i is the weight of a threat a_i, $\sum_{i=1}^{n} \omega_i = 1$, μ_{a_i} is the membership degree of a_i and v_{a_i} is the non-membership degree. The reliability of IFWA is also demonstrated because the aggregation of the membership degrees in IFWA is similar to the aggregation of the damaged probabilities of UAV, as formula 3 shows.

$$P_k = 1 - \prod_{j=1}^{k}\left(1 - P_j\right) \tag{3}$$

where P_k is the damaged probability of UAV after it has exposed in threats for k times, P_j is the damaged probability of UAV at the jth exposure.

If UAV has exposed in $k-1$ threats, the membership degree of the kth threat will increase to $\mu_k^{'} = 1 - (1 - \mu_k)\prod_{i=1}^{k-1}(1 - \mu_i)$ where μ_i is the membership degree of the ith threat.

The risk in the threats intersection area is higher than that in the equivalent area of any single threat. We also aggregate the threats in the intersection area by the IFWA operator. The weight of a threat (r_i) in the intersection area is set to be $1/m_r$ where m_r is the number of threats in the area.

3.3.1 Threats aggregation on a tree node or an edge

Figure 4 shows threats aggregation on a tree node or an edge. The violet dotted circles denote the boundaries of the distributions of UAV states at path tree nodes. We straightly use the result of LQG-MP [21] to calculate the standard variance of the apriori probabilistic distribution of the UAV states at each tree node q_i off-line. A "Threat Boundary" is decided by the maximum threat radius. The membership degree in the red area is biggest and the degree in the dark blue area is smallest. The transition of colors from red to dark blue denotes the decreasing of the degree. This color expression is used in the following figures.

A set of particles $p_j, 1 \le j \le m$ $(m = 60)$ are sampled around a tree node q_i, as the violet nodes in Fig. 4. The positions distribution of the particles $Pos(p_j) \sim \mathrm{N}(Pos(q_i), \delta_{q_i}^2)$ where $\delta_{q_i}^2$ denotes the variance of the UAV state distribution at q_i. Each particle p_j is regarded as a likely state of UAV. The threat level at the node q_i is defined as the mean of the threat levels of p_j, $\mathrm{Tl}(q_i) = \sum_{j=1}^{m} \omega_{p_j} Tl(p_j)$. The weight ω_{p_j} is set to be $1/m$.

The threats on an edge between two adjacent tree nodes are aggregated by $\frac{1}{le}\int_\xi a(\zeta, s(\zeta))\mathrm{d}_\xi$ where le is the length of the edge, ξ is the function of the edge, ζ denotes a point on ξ, d_ξ is the unit arc length, $a(\zeta, s(\zeta))$ is the threat at ζ, $s(\zeta)$ is the UAV state. The integral is not a Riemann integral because it utilizes "$\oplus$" instead of "+". To relieve the

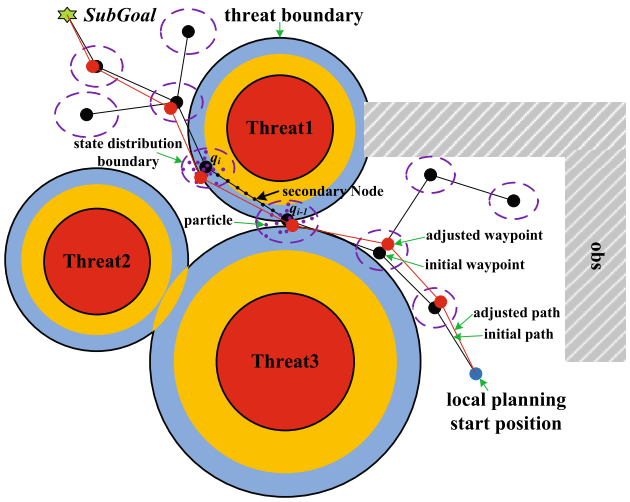


Fig. 4 Radar threat assessment and safety adjustment

computational overhead, the integral is replaced by a discretization method for the online problem. Fig. 4 shows the threats aggregation on $edge_{i-1,i}$ between q_{i-1} and q_i. $edge_{i-1,i}$ is discretized into n_q secondary nodes ($sq_j, 1 \le j \le n_q$). The threats on q_i and sq_j are aggregated by IFWA where the weights are set to be $1/(n_q + 1)$, $n_q = 10$.

4 Online path planning

In an uncertain and dynamic environment, no global path can be immediately generated based on the apriori information [46–48]. Thus, the global planning problem is subdivided into a series of simpler but practical sub tasks according to the real-time feedback information and planning situations.

4.1 Model of the online path planning algorithm

Our planning model mainly incorporates a Threat Assessor (TA) and a Path Planner (PP), as Fig. 5 shows. The distributions of UAV states are estimated by LQG-MP beforehand, to help TA and PP to deal with the uncertainties of the motion and observation of UAV. TA aims to provide quantized results of threats assessment for PP to handle threats. After a local path is searched, a safety adjustment is executed to further reduce the threat on the path. If a global path is found, the path will be optimized in the remaining planning time. The environmental information is updated and provided for TA and PP.

4.2 Sub goal selection

Our sub goal selection is inspired by the work in [48] which selects sub goals by A*. But the predefined sub goals become invalid in an uncertain and dynamic environment. Thus, we select sub goals online according to both the apriori information and the real-time planning situations. We assume: the distribution of obstacles is known; the threat information inside SHs is noise-free. We select sub goals on the boundaries of SHs and plan local paths within SHs.

The heuristic sub goal selection process is shown by formula 4. a. The boundary of the present SH is discretized into 60 points to construct a point set (SG). b. The obstacle-occupied, high risk and flight-constraints-violated points are removed from SG. If the threat level on a point exceeds a specified threshold (0.5 in this study), the point is regarded as a high risk point. c. The costs $Cost_{SG}$ of the remaining points are calculated, and the point with the minimum cost is chosen to be the present sub goal ($SubGoal$).

$$SubGoal = \min_{SG} Cost_{SG} \tag{4}$$

s.t. $(SG \subseteq \partial win(Lu)) \cap (SG - SG_{Obs} - SG_{HR} - SG_{CV})$

Lu denotes a UAV location. $\partial win(Lu)$ denotes the boundary of the present SH centering at Lu. SG_{obs} denotes the obstacle-occupied points. SG_{HR} denotes the high risk points. SG_{CV} denotes the flight-constraints-violated points. If the global goal ($Goal$) is inside the present SH, then set $SubGoal = Goal$.

$Cost_{SG}$ is defined according to the key idea of the artificial potential field method. We regard obstacles and threats as repulsion forces against path planning and $SubGoal$s as attraction forces for path planning. $Cost_{SG}$ is used to accelerate the path searching process and reduce the threat on a path, as:

$$Cost_{SG} = C_{\gamma_1} L_{ETSG} + C_{\gamma_2} L_{SGTG} + C_{\gamma_3} \sum_{i=1}^{N_R} \omega_i R_i + C_{\gamma_4} R_D \tag{5}$$

where C_{γ_k} is the weight and $\sum_{k=1}^{4} C_{\gamma_k} = 1$. We denote the line from the end of the last local path End_j to SG as $Line_{ETSG}$ and the line from the SG to $Goal$ as $Line_{SGTG}$. L_{ETSG} and L_{SGTG} partially reflect the obstacle complexities on $Line_{ETSG}$ and $Line_{SGTG}$ respectively. N_R is the number of STs intersecting with $Line_{ETSG}$. R_i is the maximum threat radius of the ith ST, $\omega_i = 1/N_R$. R_D is the estimated motion radius of DT in the shortest time UAV reached a SG. If the DT threat area does not intersect with $Line_{ETSG}$, then set $R_D = 0$, otherwise, $R_D = \sqrt{v_D(t_E + (LL_{ETSG}/v_{uav}))}$ where v_D and v_{uav} are the present velocities of DT and UAV, respectively, t_E is the earliest time UAV reached End_j, LL_{ETSG} is the length of $Line_{ETSG}$.

$$L_{ETSG} = L_{\gamma_1} LL_{ETSG} + L_{\gamma_2} \sum_{i=1}^{N_{os_1}} H_{os_i} + L_{\gamma_3} \sum_{i=1}^{N_{os_1}} W_{os_i} \tag{6}$$

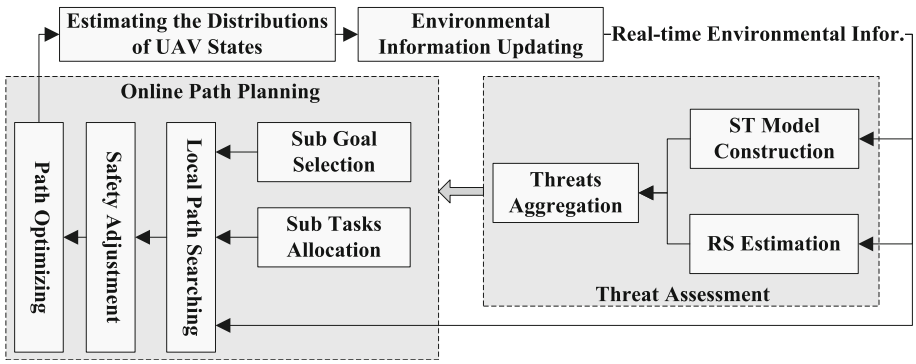


Fig. 5 Model of the online path planning algorithm

where N_{os_1} is the number of the obstacles on $Line_{ETSG}$. H_{os_i} is the height of an obstacle on $Line_{ETSG}$. W_{os_i} is the width of the projection of an obstacle in the vertical direction of $Line_{ETSG}$. L_{γ_k} is the weight, $\sum_{k=1}^{3} L_{\gamma_k} = 1$. $L_{SGTG} = L_{\gamma_1} LL_{SGTG} + L_{\gamma_2} \sum_{i=1}^{N_{os_2}} GH_{os_i} + L_{\gamma_3} \sum_{i=1}^{N_{os_2}} GW_{os_i}$ where N_{os_2} is the number of the obstacles on $Line_{SGTG}$. LL_{SGTG} is the length of $Line_{SGTG}$. GH_{os_i} is the height of an obstacle on $Line_{SGTG}$. GW_{os_i} is the width of the projection of an obstacle in the vertical direction of $Line_{SGTG}$. The minimum cuboid bounding boxes of obstacles are used to simplify the calculation of $Cost_{SG}$.

Figure 6 shows the *SubGoal*s selection process. A green cross denotes a UAV location when a local planning starts. The circles denote SH boundaries. The blue and red crosses denote the start position and *Goal* of the global planning, respectively. A star denotes a *SubGoal*. Because *Goal* is inside the seventh SH, we set $SubGoal_7 = Goal$.

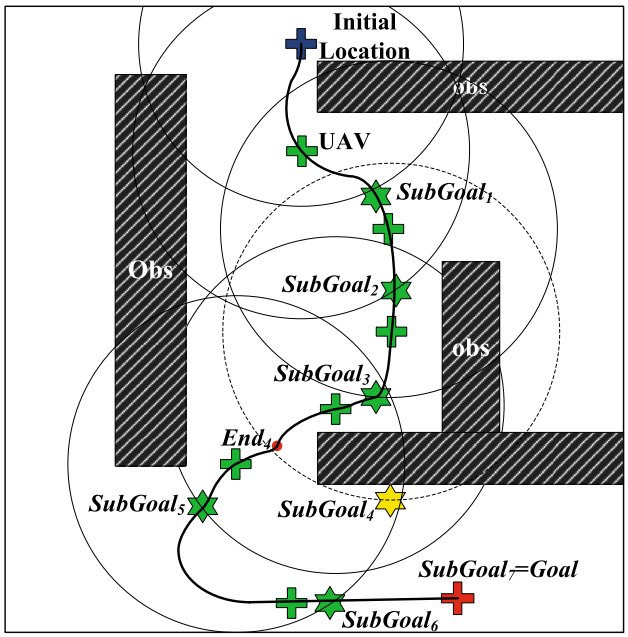


Fig. 6 Illustration of sub goals selection and sub tasks allocation

4.3 Sub tasks allocation

We improve the online planning frame in [12] for the sub tasks allocation. Time Horizon (TH) is defined as the maximum allocated time for a local planning. TH should be long enough to ensure that most local planning can find a path [12]. The definition of TH should also take the UAV motion into account to ensure that a path is planned before its execution.

A local planning task incorporates the sub tasks of path searching and safety adjustment. If no local path towards the present *SubGoal* is found in TH, the sub tasks allocator will reserve the valid intermediate result and start the next iteration based on the intermediate result. $SubGoal_4$ and the dashed-line circle in Fig. 6 show that the fourth local planning does not find a local path towards $SubGoal_4$. In this case, the valid sub path from $SubGoal_3$ to End_4 is kept and the fifth iteration is started based on the valid sub path. Our algorithm classifies path tree nodes by their TSs and neglects the nodes outside the present TH to decrease the computational overhead.

If the planning does not time out when *Goal* is reached, RRT* will optimize the path in the remaining time. The planning process combines the strategies of optimizing and feeding back. Thus, it has the advantages of both strategies.

4.4 Local path planning

4.4.1 Local path searching

UAV can effectively avoid threats by flying in the low altitude to utilize the radar blind spots and the masking of obstacles. DDRRT is efficient in obstacle-intense environments in the low altitude [15, 16]. If the extension of a node is blocked by obstacles, a dynamic domain (DD) is

added to the node to reduce the scope where the node can be extended. This means a sample outside the DD of its nearest tree node is directly deleted. In this way, the sampling space reduction guides DDRRT in refining a path to quickly avoid obstacles. To explicitly deal with threats and uncertainties, we define a novel threat and uncertainty based dynamic domain (TUDD) as follows.

$$\mathrm{TUDD}(q_i) = \min\left\{k_c p_c(q_i)\mathrm{DD}(q_i), \frac{\mathrm{DD}(q_i)}{k_T Tl(q_i)}\right\} \tag{7}$$

where $p_c(q_i)$ is the probability that UAV is obstacle-free at q_i, weighted by k_c. p_c is used to deal with the uncertainties of a vehicle's motion and observation. $\mathrm{Tl}(q_i)$ is the threat level on q_i and k_T is the influence coefficient of $\mathrm{Tl}(q_i)$.

Instead of computing p_c exactly, an approximation of p_c is applied to promote the computational efficiency [21, 22]. The approximation is reliable since a p_c is just an indicative probability that a collision is avoided. Computational efficiency is more crucial than the accuracy of p_c during the path searching process. A c_t denotes the number of the standard deviation that UAV can deviate from the candidate path before colliding with an obstacle at a q_i. A c_t is computed by the smallest distance between UAV and obstacles, and the distribution of UAV positions at a q_i. The distribution is obtained by LQG-MP off-line. For a n dimensional multivariate Gaussian distribution of UAV positions, the p_c of UAV at a q_i is approximated by $\Gamma(\mathrm{n}/2, c_t^2/2)$ where Γ is the regularized Gamma distribution function. p_c provides a lower bound of the probability of avoiding a collision at each tree node [21, 22]. The quality (Q) of a path (Π) is defined by multiplying the p_c on all the waypionts:

$$Q(\Pi) = \prod_{i=1}^{l} p_c(q_i) \tag{8}$$

A "GoalZoom" method is applied in the sampling process to promote the path searching ability of DDRRT, especially when large obstacle-free spaces exist. The "GoalZoom" method picks, with a probability (P), samples in the sphere centered at the *SubGoal* with the radius of $min_{\forall i}\mathrm{D}(q_i, SubGoal)$ where "D" denotes the Euclidean distance", q_i denotes a path tree node [13].

Once a sample (q_s) is created, its k-near tree nodes ($q_i, 1 \le i \le k$) are selected, and the node (q_n) with the minimum cost distance to q_s is chosen. If q_s falls into the TUDD of q_n, Connect$(Tree, q_n, q_s)$ will be executed. The method helps the tree to select locally optimized extensions. The cost distance between a q_i and q_s is defined as:

$$D_C(q_i, q_s) = D_{\gamma_1} D(q_i, q_s) + D_{\gamma_2} Py + D_{\gamma_3} Tl_s \tag{9}$$

where D_{γ_i} is the weight and $\sum_{i=1}^{3} D_{\gamma_i} = 1$. D denotes the Euclidean distance. A node's direction is as crucial as its location. We define it as the direction of its in-edge which ends at the node on the path tree. $0 \le Py \le \pi$ is the sum of the absolute values of angles of elevation and yaw between the directions of q_i and the line from q_i to q_s. We define that the angle of elevation will increase the path cost whereas the angle of depression will not. That guides the planner in searching paths in the low altitude. Tl_s is the threat level on the line between q_i and q_s, calculated as the threats aggregation on a tree edge.

4.4.2 Safety adjustment

If a path reaches the present *SubGoal*, the safety adjustment is executed to reduce the threat on the path. As Fig. 4 shows, the red nodes and edges are the adjusted ones. The waypoints q_i on the path are adjusted in turn. As discussed in Sect. 3.3.1, the particles p_j around a tree node q_i have been obtained.

1. The weights ω_{p_j} of particles p_j are reassigned to be $\min\left\{\frac{1}{k_T \cdot \mathrm{Tl}(p_j)}, 1\right\}$ where k_T and $\mathrm{Tl}(p_j)$ are the same as those in formula 7.
2. After normalization, the final ω_{p_j} are obtained. A particle stands for a likely UAV state under a control input. The weights reassignment adds a kind of bias to the possible control inputs to reduce the threat on a path.
3. The new state (s) of q_i is estimated by $s(q_i') = \sum_{j=1}^{N_s} \omega_{p_j} \cdot s(p_j)$. The new edge is also determined. If the collision probability of q_i' or the new edge exceeds a certain threshold (0.5 in this study) or a constraint is violated, the adjustment on q_i fails and the method will adjust the next waypoint.

4.4.3 Path optimizing

RRT* optimizes its solution towards the asymptotically optimal path over time [37, 39, 40]. After a path towards *Goal* is searched, RRT* is used to optimize the remaining part of the path which will not be executed in the remaining planning time t_r. The remaining path starts at a waypoint ($Root_r$) which is the nearest to UAV on the path while UAV cannot reach in t_r. We denote the sub tree rooting at $Root_r$ as the remaining path tree which is extended by our modified DDRRT and optimized by RRT* in t_r.

In RRT*, the upper cost bound ($\mathrm{UB}(q)$) of a tree node (q) is defined as the actual cost of the path from q to *Goal*

computed by formula 9. If no such path exists, $\mathrm{UB}(q)$ is set to be ∞. The lower bound ($\mathrm{LB}(q)$) is defined as the predicted lowest cost distance from q to *Goal*, i.e., $D_C(q, Goal)$. Once we find a new path from q to *Goal* and q is the farther node of *Goal* on the path, $\mathrm{UB}(q)$ is set to be $\mathrm{LB}(q)$ and the following iterations are executed along the remaining path. For each waypoint (q_i) and its father waypoint (q_{i-1}) on the path, if $\mathrm{UB}(q_{i-1}) > \mathrm{UB}(q_i) + D_C(q_{i-1}, q_i)$, then set $\mathrm{UB}(q_{i-1})$ to be $\mathrm{UB}(q_i) + D_C(q_{i-1}, q_i)$. Each children node ($ch_j$) of q_{i-1} except q_i on the remaining path tree will be checked. If $\mathrm{LB}(ch_j) + D_C(q_{i-1}, ch_j) > \mathrm{UB}(q_{i-1})$, the sub tree rooted at ch_j will be safely removed because it cannot provide a better solution than the newly found path. If $\mathrm{UB}(q_{i-1}) \leq \mathrm{UB}(q_i) + D_C(q_{i-1}, q_i)$, the present execution of the path optimizing method will be terminated and the sub tree rooted at q_i will be checked whether it can be removed.

RRT* generates a series of solutions to get a less costly path. The pruning of tree nodes can improve the computational efficiency of RRT* [37, 39, 40].

4.5 Algorithms implementation

Algorithm 1 is the RS estimation algorithm. "PFPrediction" is the DT motion prediction function by PF. "Update" is the *TTree* updating function. *RandNum* is a uniformly and randomly distributing number in [0,1]. "reach" and "multi-connect" are the RS searching operators. q_{dr} is a directly DT reachable node. "RSCal" aims to search for indirectly reachable nodes among the neighbors of q_{dr} on *PathTree*. "DistCal" aims to estimated the nearest distance set *DS* between DT and its reachable nodes by the DT motion state *DTState* and the predicted earliest time t_{rs} DT reaches the nodes in RS.

Algorithm 1 RS Estimation Algorithm

Require:
a new $PathTree$ node, q_{new}; a DT observation, $DTObs$; probabilistic threshold, P_{itd}; $TTree$; $PathTree$

Ensure:
$TTree$; RS; the nearest distance between DT and the nodes in RS, DS

$TNode_{new}$ = PFPrediction($DTObs$);
Update($TTree$, $TNode_{new}$, $DTObs$);
if $0 \leq RandNum \leq P_{itd}$ **then**
  [$TTree$, q_{dr}, t_{dr}] = reach(q_{new}, $TTree$);
else
  q_s=GenerateSample();
  [$TTree$, q_{dr}, t_{dr}] = multi-connect(q_s, $TTree$, $PathTree$);
end if
if $t_{dr} \neq \infty$ **then**
  [RS, t_{rs}] = RSCal($PathTree$, $TTree$, q_{dr}, t_{dr});
end if
while $RS \neq$ Null **do**
  DS = DistCal($PathTree$, $TTree$, t_{rs}, $DTState$);
end while
return ($TTree$, RS, DS);

Algorithm 2 UAV Online Path Planning Algorithm in Uncertain and Hostile Environments

Require:
environmental information, Env; planning start location, SL; $Goal$; DT start location, DL

Ensure:
$PathTree$; UAV flyable path: $Path$

$PathTree$.init(SL), $TTree$.init(DL);
while !Reach($Goal$) && $T_p \leq TH_p$ **do**
  select $SubGoal$, initialize sub task;
  while !Reach($SubGoal$) && $T_{lp} \leq TH_{lp}$ **do**
    [q_{new}, q_n]=TreeExplore(Env, $PathTree$);
    if q_{new} == null **then**
      $q_n.TUDD = \min\{DD, q_n.TUDD\}$;
    else
      for $st_i \leq NumofST$ **do**
        $q_{new}.ST(st_i)$ = TtCal($PathTree$, $Dist(q_{new}, ST(st_i))$, $ST(st_i).Model$, $q_{new}.UavState$);
      end for
      [$TTree$, RS, DS] = RSE(q_{new}, $DTObs$, P_{itd}, $TTree$, $PathTree$);
      $RS.DT$ = TtCal($PathTree$, DS, $DTModel$, $RS.UavState$);
      $RS.Threat$ = IFWA($RS.ST$, $RS.DT$);
      $q_{new}.TUDD$ = TUDDCal(p_c, $q_{new}.Threat$);
      update Env and $PathTree$;
    end if
  end while
  $SubPath$ = FindPath($PathTree$, $SubGoal$);
  $SubPath$ = SafetyAdjustment(Env, $PathTree$, $SubPath$);
  $Path = Path + SubPath$;
end while
while $T_p \leq TH_p$ **do**
  $Root_r$ = FindFreePathNode($Path$, $UavState$, T_p);
  [$PathTree$, $Path$] = RRT*($PathTree$, Env, $Root_r$);
end while
if !reach($Goal$) **then**
  Emergency Strategy;
end if
return ($PathTree$, $Path$);

Algorithm 2 shows the pseudocode of our path planning method. T_p is the duration of the present planning and TH_p is the allocated total planning time. T_{lp} is the duration of the present local planning and TH_{lp} is the TH. "TreeExplore" is the improved heuristic DDRRT for the *PathTree* extension. q_n indicates an unsuccessfully extended *PathTree* node. DD denotes the predefined dynamic domain. "TtCal" aims to calculate the threat level on a node. "IFWA" is the threats aggregation function. "RSE" denotes the RS estimator introduced in algorithm 1. "TUDDCal" aims to calculate the TUDD subject to formula 7. In the *PathTree* updating process, the TSs of all the *PathTree* nodes are updated. "SafetyAdjustment" is the safety adjustment function. "FindFreePathNode" aims to find $Root_r$, as discussed in Sect. 4.4.3. The remaining part of the path is optimized in $TH_p - T_p$ by RRT*.

4.6 Algorithm analysis

The result of RRT or DDRRT does not converge to the optimal path [1–3, 15, 16]. RRT* optimizes a path towards the asymptotically optimal one over time, by improving the topological structure of the path tree [40]. Our method

avoid threats explicitly in the path searching process under uncertainties. It reduces the threat on a local path by the safety adjustment. If a path towards *Goal* is found, it is optimized by RRT* in the remaining planning time. Therefore, our result is an improved one without the guarantee of the optimization.

The randomized path planning methods do not have any running time upper-bound, i.e., a complete plan is not surely returned within an acceptable time [40]. Let n be the number of samples. The asymptotic time complexities of both RRT and RRT* are $O(n \log n)$ in terms of the number of simple operations, such as comparisons, additions, and multiplications [40]. Collision Detection (CD) is a time-costly high-level function in a RRT-based method [1, 15, 16]. DDRRT reduces the number of CD called by the sampling space reduction to accelerate its path searching process [15, 16]. Our method adopts a similar sampling space reduction method to DDRRT in the path searching process. Thus, the time complexity of our path searching process is not higher than $O(n \log n)$. The time complexity of our path planning method is not higher than $O(n \log n)$.

RRT, RRT* and DDRRT are probabilistically complete [1, 15, 16, 40]. The probabilities that the planners fail to return a solution, if one exists, decay to zero as the number of samples approaches infinity. Our method is still probabilistically complete since it does not destroy this completeness. Like RRT-reach, our RS estimator has no guarantee of the probabilistic completeness for the sake of computational efficiency. As discussed in Sect. 2.2, our method has advantage in planning under complex and dynamic constraints.

5 Simulation results

5.1 Simulation environments setting

We perform experiments to verify that our algorithm can be applied online in an uncertain and hostile environment. The RS estimator is executed in parallel with the UAV path planner. We make 100 Monte Carlo simulations for each method in each scenario. The experiments are done on computers with Intel Pentium 4, 2.5 GHz CPU, 1GB RAM, Windows XP OS and Matlab R2010a.

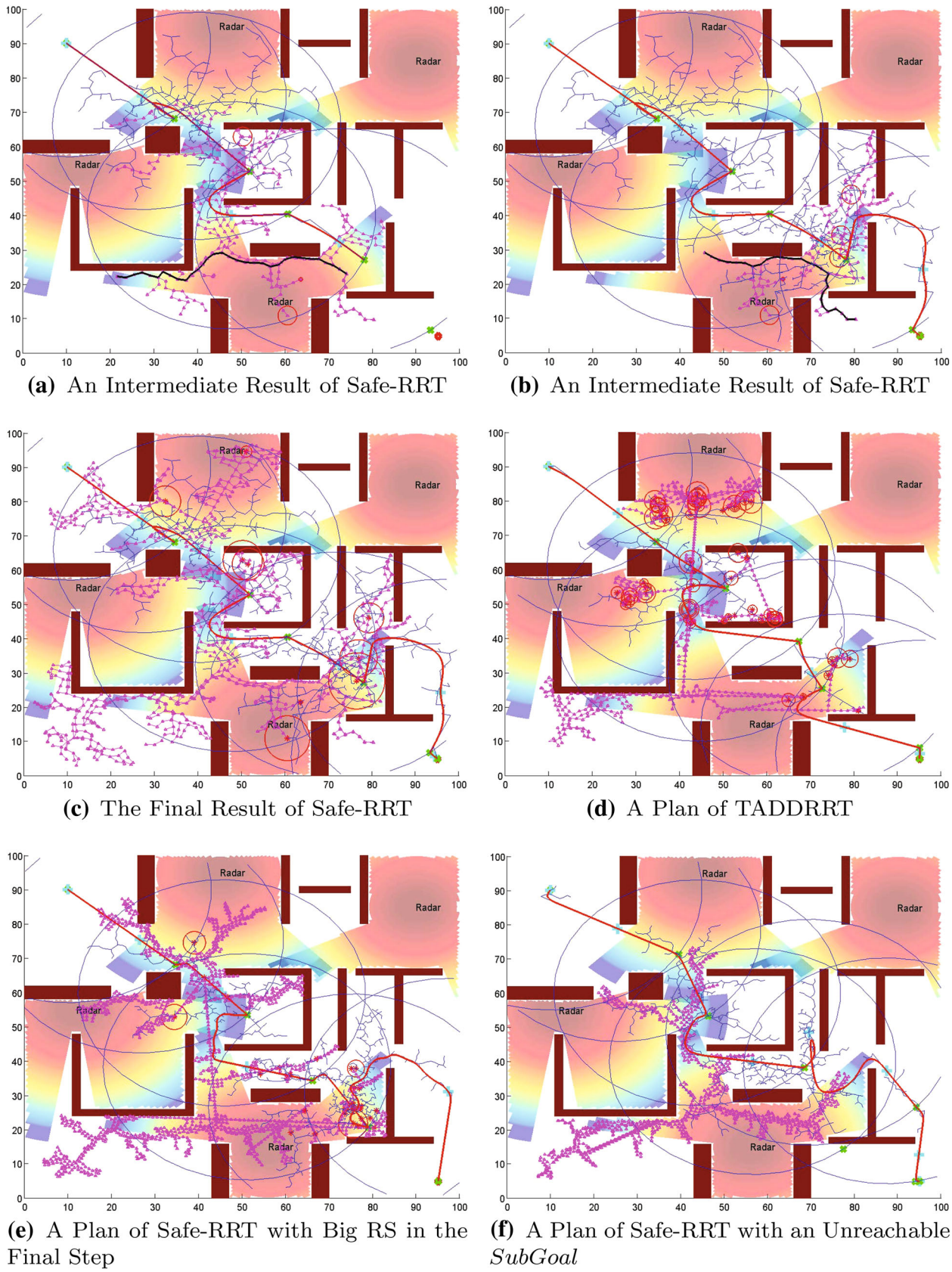
The environments are created according to the real world planning system. The duration of one path tree extension step $\tau = 1$ second. One extension step length $le = v \cdot \tau$ where v is the velocity of a vehicle. The radius of DD is set to be an optimal value $10 \cdot le$ [15, 16]. In the 2D environment, we set the UAV motion parameters as: $v_{min} = 0.5$ units/s, $v_{max} = 1.5$ units/s, $a_v = 0.02$ units/s^2, $r_{min} = sl_{min} = 5$ units; the DT velocity v_D = 1 unit/s; TH= 15 seconds; UAV starts at [10, 90], ends at [95, 5]; DT starts at [10,10]; the maximum threat radius of radar $R_{max} = 38$ units; the maximum static threat radius of DT $RD_{max} = 10$ units; the radius of SH is 33 units; the v_p in the non-membership function is 1 unit/s. In the 3D environment, the motion parameters of UAV and DT are set to be five times of those in the two dimensional scenario; the radar characteristic coefficient $K_R = 0.003$; TH=25 seconds; UAV starts at [950, 80 ,80], ends at [50, 950, 100]; DT starts at [950,950,0]; R_{max} = 380 units; $RD_{max} = 50$ units; v_p = 5 units/s; the radius of SH is 150 units.

The heuristic parameters are set as follows. We use the asymptotic k-near neighbors searching method in [40], the lower bound of k is set to be 3. In the GoalZoom heuristics, $P = 0.3$. In Algorithm 1, $P_{itd} = 0.5$. In formula 2, $\omega_N = 0.7$, $\omega_\delta = 0.3$, $\omega_C = 1$. In formula 5, $C_{\gamma_1} = C_{\gamma_3} = 0.3$, $C_{\gamma_2} = C_{\gamma_4} = 0.2$. In formula 7, $k_c = 1$, $k_T = 5$. In formula 9, $D_{\gamma_1} = 0.5$, $D_{\gamma_2} = 0.1$, $D_{\gamma_3} = 0.4$. The weights of the formula 6 in the 2D scenario are set as: $L_{\gamma_1} = 0.7$, $L_{\gamma_2} = 0$, $L_{\gamma_3} = 0.3$; in the 3D scenario: $L_{\gamma_1} = 0.7$, $L_{\gamma_2} = L_{\gamma_3} = 0.15$.

5.2 Planning results

The elements in Figs. 7 and 8 are interpreted as follows. The blue and red diamonds denote the start location and the *Goal* of planning, respectively. The red curve indicates a UAV path smoothed by the Dubins curve. A colored area denotes a radar threat area masked by obstacles. The blue-branch and magenta-branch trees denote the UAV and the DT path trees, respectively. A red star denotes a directly DT reachable node. A red circle centering at a red star denotes an estimated DT reachable range which includes both the directly and indirectly DT reachable nodes. A green star denotes a *SubGoal*. A blue circle denotes a SH boundary in the 2D scenario. A cyan cross denotes a UAV location when selecting a *SubGoal* in the 3D scenario. The obstacle environments derive from the puzzles of Bug Trap and Labyrinth [1, 2, 15, 16] which can test a sampling based planning algorithm. It is rather complicated due to the following reasons: the environment includes narrow hallways and traps composed by complex obstacles (convex and nonconvex); it invalidates the traditional Euclidean distance based heuristics. Moreover, the planning is required to deal with uncertainties and avoid threats especially DT. Meanwhile, it must plan paths online under limited and real-time threat information.

Figure 7 shows the simulation results in a 2D environment. Figure 7 verifies that our Safe-RRT is effective, because of the following reasons: the UAV path tree is inclined to search low risk path in the low-threat or threat-free areas; the result path keeps an appropriate distance from obstacles to avoid collision. The high path searching



(a) An Intermediate Result of Safe-RRT

(b) An Intermediate Result of Safe-RRT

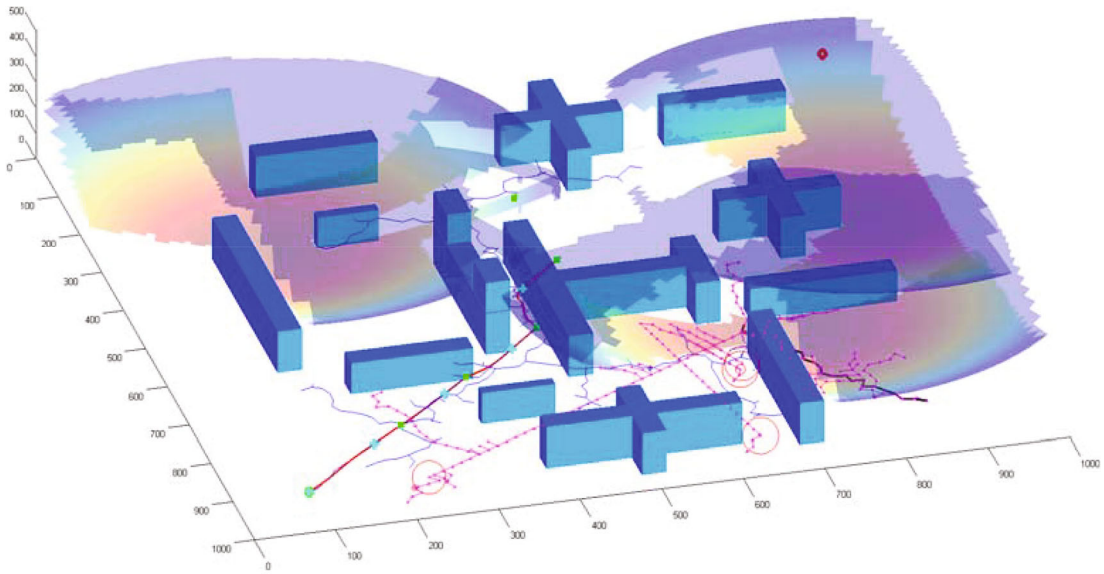
(c) The Final Result of Safe-RRT

(d) A Plan of TADDRRT

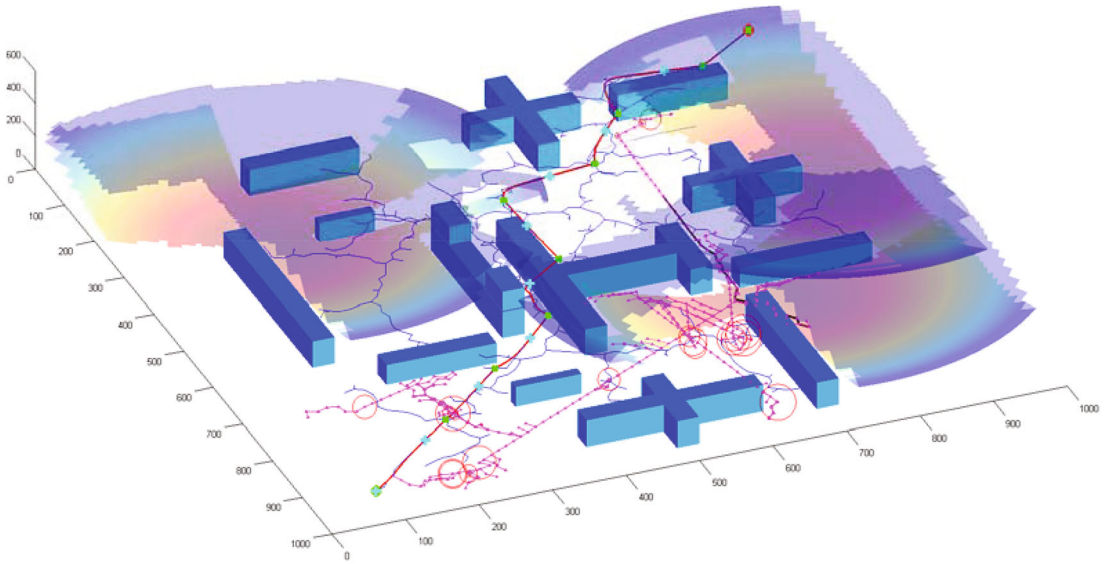
(e) A Plan of Safe-RRT with Big RS in the Final Step

(f) A Plan of Safe-RRT with an Unreachable *SubGoal*

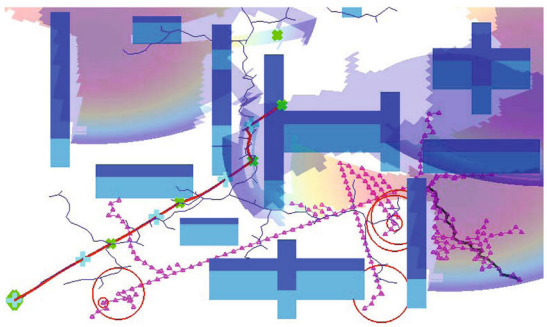
Fig. 7 The two dimensional simulation results



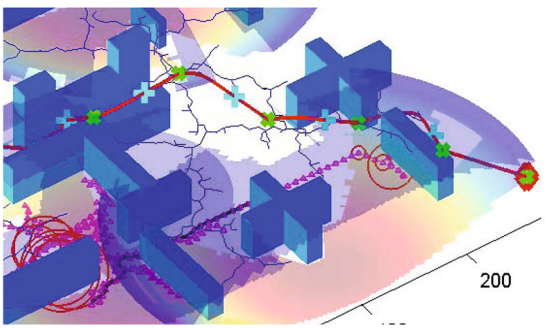
(a) Result at the One-fourth Time of a Plan



(b) Result at the Three-fourths Time of the Plan



(c) DT pursuing UAV



(d) UAV avoiding DT by utilizing the masking of obstacles

Fig. 8 The three dimensional simulation results

ability of our Safe-RRT in obstacle-complex environments is also verified. That is because the result path is able to get through narrow hallways with a small number of tree branches. The ability can benefit the threats avoidance ability of Safe-RRT by utilizing the masking of obstacles to threats. The planner also avoids DT by utilizing the masking of obstacles, e.g., it utilizes obstacle in the lower right corner of Fig. 7b to avoid DT.

Figure 7a, b, c show the intermediate and final results of Safe-RRT. They show both the process of UAV path planning and the updating processes of DT path tree and DT path. The sub goal selections conform to formula 4. The black lines in Fig.7a and b denote the DT pursuing path which is searched online. DT is not required to finish the whole path due to its limited velocity. Our DT motion simulation satisfies the requirement of the DT avoidance model in Sect. 2.3, because: the DT path is flexible and intelligent with a strong bias of threatening UAV; DT tries to pursue the UAV path tree nodes by diversely extending its path tree. Meanwhile, the large DT path tree can better estimate the RS to assess the DT threat from the UAV path planning point of view.

Figure 7d shows a plan of the Threat Assessment based DDRRT method (TADDRRT). TADDRRT has the same planning process as our algorithm. It is the reimplementation of DDRRT in hostile environments. It does not consider the uncertainties of UAV motion and observation. Its result path is closer to obstacles causing a higher collision probability than that of Safe-RRT. Thus, our consideration of the uncertainties is appropriate. If we pre-expand obstacles as traditional methods do, albeit the path can keep away from obstacles, the narrow hallways may be blocked by the expansion. TADDRRT has no heuristics of "GoalZoom", "k-neighbors" and "RRT*". Meanwhile, it underestimate the certainty degrees of threats, further underestimate threats, because it sets the standard deviations of the distributions of UAV states to be zero in formula 2. Thus, the result of TADDRRT is more inclined to get through high risk areas near threat centers than those of Safe-RRT. Thus, the result path of TADDRRT is more risky than those of Safe-RRT. That testifies our improvements based on TADDRRT are effective. Figure 7e shows the last part of path is zigzag. That is because the RS in the last iteration is big, Safe-RRT creates more tree nodes to avoid DT more explicitly. That also testifies the good threats avoidance ability of Safe-RRT. Figure 7f shows that the fourth *SubGoal* in the lower right corner is not reached in TH. Safe-RRT starts the fifth local planning based on the result of the fourth iteration.

Figure 8 shows our planning results in a 3D environment. The good abilities of path searching, threats avoidance and uncertainties processing of our planner are certified. The effectiveness of our RS estimator is also verified.

Figure 8a and b are two intermediate results of our planner at the one-fourth and three-fourths planning time of a plan, showing the processes of UAV path planning and DT pursuing UAV. The paths can keep appropriate distances from obstacles. That verifies our dealing with the uncertainties of UAV motion and observation is effective. The good threats avoidance ability of Safe-RRT is also certified by Fig. 8a and b. That is because of the following reasons: the paths can avoid radar threats by utilizing the radar blind spots and the masking of obstacles to threats in the low altitude; there are many DT reachable areas in Fig. 8b, however, Safe-RRT can still find some relatively safe areas to avoid threats. Figure 8a and b show that the DT path tree grows towards the UAV path tree with strong bias. DT can flexibly and intelligently adjust its path to pursue UAV. The observations verify the DT motion simulation is reliable and our RS estimator is effective.

Figure 8c is a detail of Fig. 8a, showing the DT pursuing UAV by enlarging its path tree. That certifies the reliability of our DT motion simulation and the effectiveness of the RS estimation. Figure 8c also shows our planner can find a path in the narrow hallway in the middle of Fig. 8a with a small amount of branches. That verifies the good path searching ability of our planner. Figure 8d is a detail of Fig. 8b, showing that UAV avoids DT by the masking of obstacles. The red circles denote the DT reachable ranges. Fig. 8d especially shows that the planner can utilize the masking of the obstacle at the top right of Fig. 8b to avoid DT. That verifies our DT avoidance method is effective.

5.3 Results and analyses

The comparisons are made between Safe-RRT and TADDRRT, NFZ-DDRRT, RRT*, as the statistical results in Table 1. The TUDDs of the nodes in RS are set to be zero in NFZ-DDRRT according to the discussion in Sect. 1.2. NFZ-DDRRT is reimplemented with the same ST avoidance ability as TADDRRT. All approaches share our planning framework.

PST is the path searching time. NCDC is the number of collision detection called in the path searching process. ESR = NTN/NCDC is the extension success rate of path tree nodes. NTN is the number of the UAV path tree nodes. TSI is the time of one path searching iteration, TSI = PST/NCDC. PST, NCDC and TSI reflect the time complexity of the path searching phase of a RRT-based algorithm. ESR reflects the reliability of the heuristic sampling space reduction to some extent. FR is the failure rate, an execution fails if no path towards *Goal* is returned in the available planning time which is set to be 80 and 160 s in the 2D and 3D scenarios, respectively. PC is the path cost computed by formula 9. Q(Π) is the path quality calculated by formula 8. PST, NCDC, ESR and TSI are the technical indicators in

Table 1 Comparisons of Safe-RRT with the traditional methods

Dim	Algorithm	PST(s)	NCDC	ESR	TSI(s)	Q(Π)	PC	FR
2D	Safe-RRT	7.92	1156	0.5087	0.0069	0.2474	135.43	0.033
	TADDRRT	9.51	1606	0.5367	0.0059	0.1049	149.94	0.045
	NFZ-DDRRT	11.54	1913	0.4135	0.0060	0.0677	179.91	0.064
	RRT*	14.11	2662	0.4189	0.0053	0.0661	215.33	0.055
3D	Safe-RRT	13.57	1213	0.5441	0.0112	0.3571	1565.21	0.013
	TADDRRT	13.39	1313	0.5453	0.0102	0.2255	1635.05	0.011
	NFZ-DDRRT	16.38	1906	0.4654	0.0086	0.1633	1883.22	0.087
	RRT*	23.28	4477	0.3761	0.0052	0.1422	1962.63	0.081

Table 2 Comparisons on different subgoal selecting methods

Dim	Method	PST(s)	NTN	NCDC	ESR	TTA	PC	FR
2D	HM	7.92	588	1156	0.5087	33.23	135.43	0.033
	EDBM	11.07	710	1628	0.4361	39.17	167.73	0.082
3D	HM	13.57	660	1213	0.5441	60.59	1565.21	0.013
	EDBM	19.49	784	1746	0.4490	75.26	1728.38	0.053

the path searching process. Other technical indicators are collected after a whole planning ends.

The PST and NCDC of our Safe-RRT are the lowest in both the 2D and 3D environments, verifying its best path searching ability. That also verifies the effectiveness of our heuristics in Safe-RRT comparing to TADDRRT. The ESRs of Safe-RRT and TADDRRT are higher than the other two methods, demonstrating our sampling space reduction is effective. The TSI of Safe-RRT is higher than TADDRRT because Safe-RRT devotes extra time to handle uncertainties. The highest TSI of Safe-RRT also contributes to the best Q(Π). The PC of Safe-RRT is the best. The PC of TADDRRT is inferior to Safe-RRT, because: it is less heuristic than Safe-RRT; its path is more risky than Safe-RRT because it underestimates threats, as discussed in Sect. 5.2. The PC of NFZ-DDRRT is far worse than TADDRRT because it tries to totally detour from DT causing a longer and possibly more risky path. RRT* has the worst PC because its optimizing time is not satisfied. The better FRs certify Safe-RRT and TADDRRT are far more robust than the other two methods.

Table 2 compares our Heuristic sub goal selection Method (HM) with the traditional Euclidean Distance Based Method (EDBM). The difference between HM and EDBM derives from the principle described in formula 5: HM not only takes the distances into account, but also considers the problems of obstacles avoidance and threats avoidance, EDBM just considers the distances. Both methods are integrated into our path planning system. TTA is the total radar threat amount calculated by adding the radar membership degrees on all the waypoints and secondary waypoints on a path. We discretize a sub path between two adjacent waypoints into 10 secondary waypoints when computing TTA. HM has lower PST, NTN, NCDC, TTA, PC and FR than EDBM, verifying that our sub goals can lead the planner rapidly finding a low cost and low threat path while avoiding the local optimum to some extent. The higher ESR demonstrates HM is able to select sub goals in less obstacle-occupied spaces. Thus, HM is more informative and effective than EDBM.

Table 3 Comparisons on the static threats avoidance abilities of various algorithms

Dim	Algorithm	ATP	TTA	NHTW
2D	Safe-RRT	0.1211	33.23	4
	TADDRRT	0.1976	35.79	6
	RRT*	0.2536	47.54	13
3D	Safe-RRT	0.0821	60.59	15
	TADDRRT	0.1313	85.59	22
	RRT*	0.1675	121.51	49

Table 3 compares the ST avoidance ability of Safe-RRT with those of TADDRRT and RRT*. ATP is the aggregated radar threat on the result path. It is calculated by aggregating the radar threats on all waypoints on the path using IFWA. NHTW is the number of high radar threat waypoints of which the threat levels exceed 0.5. Our method has less ATP, TTA and NHTW than TADDRRT, verifying our heuristic improvements can promote the ST avoidance ability. The result of RRT* is far inferior to those of Safe-RRT and TADDRRT, because its long optimizing time is not satisfied.

Our Extended RRT based RS estimator (ExRRT) is compared with RRT-Reach and RRT in Table 4. We suppose RRT-Reach [33] is provided with the same information

Table 4 Comparisons of various DT reachability set estimators

Dim	Method	ADTP	TDTA	ETDTA	ETDTA/TDTA	TFP/PD	RRS
2D	ExRRT	0.0855	22.97	19.31	0.8407	0.1883	46
	RRT-Reach	0.1245	24.39	14.05	0.5761	0.2672	34
	RRT	0.1746	34.55	11.42	0.3305	0.3558	19
3D	ExRRT	0.0622	43.26	36.39	0.8412	0.1859	72
	RRT-Reach	0.1076	50.66	39.48	0.7793	0.2287	65
	RRT	0.1468	60.19	31.50	0.5233	0.3206	57

Table 5 Comparison of the A-IFS based ST model with the probabilistic model

Dim	Model	PST(s)	NCDC	PC	FR	PSR	Q(Π)
2D	A-IFS	7.92	1156	135.43	0.033	0.8187	0.2474
	probabilistic	8.64	1262	151.34	0.035	0.7782	0.1116
3D	A-IFS	13.57	1213	1565.21	0.013	0.8726	0.3571
	probabilistic	17.23	1469	1672.16	0.013	0.8135	0.2654

of DT motion prediction as ExRRT. The probabilistic threshold, determining the execution probabilities of the modes of exploration and pursuit, is set to be 0.5 in RRT-Reach as ExRRT. RRT is totally random. The results of the three estimators are applied to our planning system.

ADTP is the aggregated actual DT threat on the UAV path calculated in the same way as ATP. TDTA is the total actual DT threat amount on the UAV path calculated in the same way as TTA. ETDTA is the estimated total DT threat amount on the UAV path. ETDTA/TDTA reflects the reliability and accuracy of a RS estimator. TFP is the time that a UAV path tree node is first pursued by DT. PD is the path duration. It is meaningless to simply compare TFP, since the PDs of different methods vary a lot. Therefore, TFP/PD is applied to reflect the effectiveness of an estimator. RSS is the RS size. Because the RS estimation is from the UAV path planning point of view, an earlier TFP/PD or a bigger RSS indicates the higher effectiveness of an estimator.

ExRRT contributes to less ADTP and TDTA, verifying it is more effective in helping the planner to avoid DT. The biggest ETDTA/TDTA verifies that ExRRT is the most effective and accurate. The earliest TFP/PD and the largest RSS prove that ExRRT is the most effective. ExRRT is superior to the RRT-Reach and far better than RRT in both environments.

Table 5 compares our A-IFS based threat model with the probabilistic model. In the probabilistic model, the nonmembership degree is set to be zero and the certainty value is set to be one. PSR is the penetration success rate computed by aggregating the damaged probabilities of UAV on all the waypoints by formula 3. All the two models are integrated into our planning system. The less PST and NCDC verify that our model can accelerate the path searching process. That is because the functions of nonmembership and certainty can help the path tree to quickly pass through threat areas. The probabilistic model is inclined to guide the path tree in detouring from threats. That causes the longer result path and the longer path planning time. That further results in a bigger PC. The higher PSR verifies that our model is more effective in helping the planner to avoid threats. The better Q(Π) testifies the advantage of our consideration of uncertainties of UAV motion and observation in the A-IFS based model. Accordingly, our model is more suitable for planning a path in an uncertain and hostile environment.

6 Conclusions

To deal with the uncertainties of environmental threats and to ensure the safety of the low altitude flight, we propose an online safe path planning algorithm for UAV. We consider the uncertainties of threats and vehicles' motions and observations during the planning process. A threat model is constructed based on A-IFS. It is verified to be more effective than the probabilistic one to describe an uncertain threat. The RS estimator of an uncertain DT is constructed based on the RRT extension and the particle filter prediction. Comparing with the traditional approaches, it is a more accurate method which is able to estimate more possible paths of DT with the intention of intercepting UAV. An online path planning method is proposed to avoid threats and obstacles when both the UAV motion and observation is uncertain. Our sub goal selection method is verified to be more effective than EDBM in accelerating the path searching process and creating a low cost path. Our planer is verified to be able to avoid threats and obstacles under uncertainties more effectively than the traditional methods. The major contributions are as follows:

1. A static threat model and a threat assessment method are proposed based on A-IFS to deal with the uncertainty of a threat.
2. The DT intention is taken into account in designing the RS estimator of DT to deal with the uncertainties of the DT motion and observation.
3. A fast online path planning algorithm is proposed to avoid threats and obstacles when both the UAV motion and observation is uncertain.

The future work is to solve the UAVs cooperation problem.

References

1. LaValle SM (2006) Planning algorithms. Cambridge University Press, Cambridge
2. LaValle SM, Kuffner JJ (2001) Randomized kinodynamic planning. Int J Robotic Res 20(5):378–400
3. Desaraju VR (2010) Decentralized path planning for multiple agents in complex environments using rapidly-exploring random trees. Ph.D. dissertation, Massachusetts Institute of Technology
4. Fiorini P, Shiller Z (1998) Motion planning in dynamic environments using velocity obstacles. Int J Robot Res 17:760–772
5. Miller B, Stepanyan K, Miller A, Andreev M (2011) 3D path planning in a threat environment. In: Proceedings of the 50th IEEE Conference on Decision and Control and European Control Conference (CDC-ECC). Orlando, USA pp 6864–6869
6. Anderson SJ, Peters SC, Pilutti TE, Iagnemma K (2010) An optimal-control-based framework for trajectory planning, threat assessment, and semi-autonomous control of passenger vehicles in hazard avoidance scenarios. Int J Vehicle Auton Syst 8(2):190–216
7. Gonsalves P, Cunningham R, Ton N, Okon D (2000) Intelligent threat assessment processor (ITAP) using genetic algorithms and fuzzy logic. In: Proceedings of the Third International Conference on Information Fusion. Paris, France pp 11–18
8. Yang G, Kapila V (2002) Optimal path planning for unmanned air vehicles with kinematic and tactical constraints. In: Proceedings of the 41st IEEE Conference on Decision and Control. Las Vegas, USA pp 1301–1306
9. Pan Su, Li Yan, Li Yingjie, Shiu Simon Chi-Keung (2013) An auto-adaptive convex map generating path-finding algorithm: genetic convex. Int J Mach Learn Cybern 4(5):551–563
10. Zhongxing M, Huibin W, Xu M, Dai P (2014) Evaluation of path stretch in scalable routing system. Int J Mach Learn Cybern. doi:10.1007/s13042-014-0285-6
11. Chang WL, Zeng D, Chen RC, Guo S (2013) An artificial bee colony algorithm for data collection path planning in sparse wireless sensor networks. Int J Mach Learn Cybern. doi:10.1007/s13042-013-0195-z
12. Petti S, Fraichard T (2005) Safe motion planning in dynamic environments. In: Proceedings of the 2005 IEEE/RSJ International Conference on Intelligent Robots and Systems. Edmonton, Canada pp 2210–2215
13. Urmson C, Simmons RG (2003) Approaches for heuristically biasing RRT growth. In: Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems. Las Vegas, USA pp 1178–1183
14. Lee J, Pippin C, Balch T (2008) Cost based planning with RRT in outdoor environments. In: Proceedings of the IEEE/RSJ international conference on intelligent robots and systems. Nice, France pp 684–689
15. Yershova A, Jaillet L, Simeon T, LaValle SM (2005) Dynamic-domain RRTs: Efficient exploration by controlling the sampling domain. In: Proceedings of the IEEE International Conference on Robotics and Automation. Barcelona, Spain pp 3856–3861
16. Jaillet L, Yershova A, La Valle SM, Simon T (2005) Adaptive tuning of the sampling domain for dynamic-domain RRTs. In: IEEE/RSJ International Conference on Intelligent Robots and Systems. Edmonton, Canada pp 2851–2856
17. Aoude G, Joseph J, Roy N, How J (2011) Mobile agent trajectory prediction using Bayesian nonparametric reachability trees. In: Proceedings of the AIAA Infotech@ Aerospace. AIAA, St. Louis pp 1587–1593
18. Koenig S, Simmons R (1998) Xavier: a robot navigation architecture based on partially observable markov decision process models. Artificial Intelligence Based Mobile Robotics, Case Studies of Successful Robot Systems, pp 91–122
19. Bailey T, Durrant-Whyte H (2006) Simultaneous localization and mapping (SLAM). IEEE Robotic Automat Mag 13(3):108–117
20. Huynh VA, Roy N (2009) icLQG: combining local and global optimization for control in information space. In: Proceedings of the IEEE International Conference on Robotics and Automation. Kobe, Japan pp 2851–2858
21. Van Den Berg J, Abbeel P, Goldberg K (2011) LQG-MP: Optimized path planning for robots with motion uncertainty and imperfect state information. Int J Robot Res 30(7):895–913
22. Cheng P (2005) Sampling-based motion planning with differential constraints. Ph.D. dissertation, University of Illinois
23. Shkolnik A, Walter M, Tedrake R (2009) Reachability-guided sampling for planning under differential constraints. In: Proceedings of the IEEE International Conference on Robotics and Automation. Kobe, Japan pp 2859–2865
24. Jaillet L, Hoffman J, Van den Berg J, Abbeel P, Porta JM, Goldberg K (2011) Eg-rrt: Environment-guided random trees for kinodynamic motion planning with uncertainty and obstacles. In: Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems. San Francisco, USA pp 2646–2652
25. Melchior NA, Kwak JY, Simmons R (2007) Particle RRT for path planning in very rough terrain. In: proceedings of the conference on NASA Science Technology. Roma pp 1617–1624
26. Pepy R, Lambert A (2006) Safe path planning in an uncertain-configuration space using RRT. In: Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems. Beijing, China pp 5376–5381
27. Burns B, Brock O (2007) Sampling-based motion planning with sensing uncertainty. In: Proceedings of the IEEE International Conference on Robotics and Automation. Roma pp 3313–3318
28. Fraichard T, Mermond R (1998) Path planning with uncertainty for car-like robots. proceedings of the IEEE International Conference on Robotics and Automation. Leuven, Belgium pp 27–32
29. Guibas LJ, Hsu D, Kurniawati H, Rehman E (2009) Bounded uncertainty roadmaps for path planning. Algorithm Found Robot VIII 57:199–215
30. Hanson ML, Sullivan O, Harper KA (2001) On-line situation assessment for unmanned air vehicles. In: Proceedings of the FLAIRS Conference. Florida pp 44–48
31. Kabamba PT, Meerkov SM, Zeitz FH (2005) Optimal UCAV path planning under missile threats. World Congress 1: 2002–2008
32. Aoude GS (2011) Threat assessment for safe navigation in environments with uncertainty in predictability. Ph.D. dissertation, Massachusetts Institute of Technology
33. Aoude GS, Luders BD, Joseph JM, Roy N, How JP (2013) Probabilistically safe motion planning to avoid dynamic obstacles with uncertain motion patterns. Autonom Robot 35:51–76
34. Aoude GS, Luders BD, How JP, Pilutti TE (2010) Sampling-based threat assessment algorithms for intersection collisions

involving errant drivers. In: proceedings of the IFAC Symposium on Intelligent Autonomous Vehicles. Lecce, France
35. Aoude GS, Luders BD, Lee KK, Levine DS, How JP (2010) Threat assessment design for driver assistance system at intersections. In: Proceedings of the 13th International IEEE Conference on Intelligent Transportation Systems. Funchal, Portugal pp 1855–1862
36. Aoude GS, Luders BD, Levine DS, How JP (2010) Threat-aware path planning in uncertain urban environments. In: Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems. Taipei, pp 6058–6063
37. Frazzoli E, Dahleh MA, Feron E (2002) Real-time motion planning for agile autonomous vehicles. J Guidance Control Dyn 25(1):116–129
38. Phillips M, Likhachev M (2011) Sipp: Safe interval path planning for dynamic environments. In: Proceedings of the IEEE International Conference on Robotics and Automation. Shanghai pp 5628–5635
39. Karaman S, Frazzoli E (2010) Optimal kinodynamic motion planning using incremental sampling-based methods. In: Proceedings of the 49th IEEE Conference on Decision and Control. Georgia pp 7681–7687
40. Karaman S, Frazzoli E (2011) Sampling-based algorithms for optimal motion planning. Int J Robot Res 30(7):846–894
41. Isaacs R (2012) Differential games: a mathematical theory with applications to warfare and pursuit, control and optimization. Courier Dover Publications, New York
42. Ehtamo H, Raivio T (2001) On applied nonlinear and bilevel programming or pursuit-evasion games. J Optim Theory Appl 108(1):65–96
43. Zeshui Xu (2007) Intuitionistic fuzzy aggregation operators. IEEE Trans Fuzzy Syst 15(6):1179–1187
44. Ye Wen, Fan Hongda, Zhu Aihong (2011) Mission Planning for Unmanned Aerial Vehicles. National Defense Industry Press, BeiJing
45. Carpenter J, Clifford P, Fearnhead P (1999) Improved particle filter for nonlinear problems. IEEE Proceedings-Radar, Sonar and Navigation 146(1): 2–7
46. Hsu D, Kindel R, Latombe JC, Rock S (2002) Randomized kinodynamic motion planning with moving obstacles. Int J Robot Res 21(3):233–255
47. Kim Y, Gu D-W, Postlethwaite I (2008) Real-time path planning with limited information for autonomous unmanned air vehicles. Automatica 44(3):696–712
48. Seok JH, Oh C, Lee JJ, Lee HJ (2011) Integrated path planning for a partially unknown outdoor environment. In: Proceedings of the IEEE/SICE International Symposium on System Integration. Kyoto pp 643–648